AN3954
Application note

Developing your own Visual Basic or C/C++ application on a DEMO-CR95HF-A demonstration board

Introduction

This application note explains how to develop an application using either Visual Basic or C/C++ language to drive the DEMO-CR95HF-A demonstration board from a host computer.

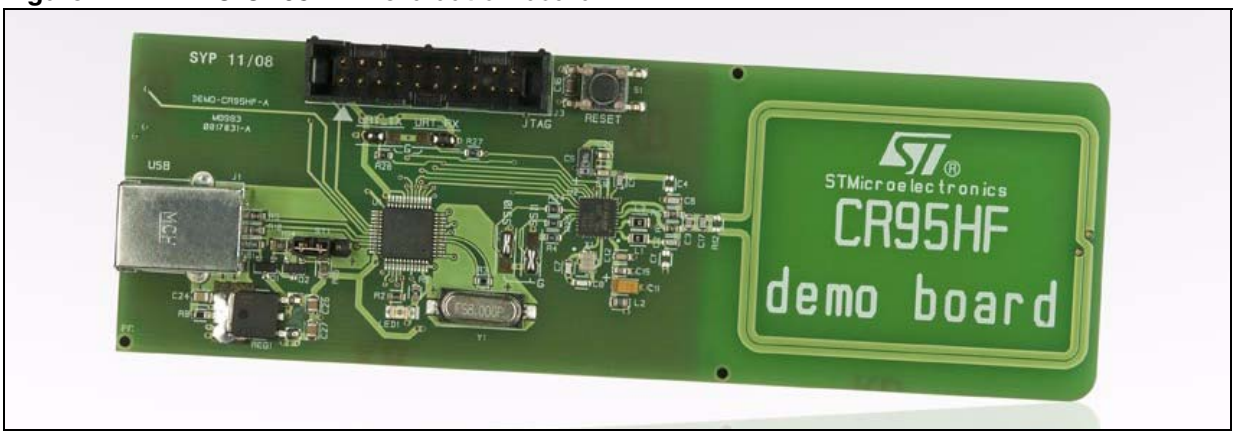
The DEMO-CR95HF-A is a demonstration kit which allows to evaluate the performance of STMicroelectronics CR95HF 13.56 MHz multiprotocol contactless transceiver. It includes a ready-to-use board to interface with the CR95HF host PC demonstration software through a USB interface.

The DEMO-CR95HF-A board is powered through the USB port and no external power supply is required. It is made up of a CR95HF contactless transceiver, a 48 x 34 mm 13.56 MHz inductive etched antenna and its associated tuning components. The CR95HF communicates with the STM32F103CB 32-bit core MCU via the SPI bus.

A dynamic link library (DLL) file is used by the host computer to manage several functions and communicate with the STM32 MCU and the CR95HF IC. It is included in the demo-cr95hf-a_dev_appli.zip version 2 (Visual Basic 6 and C/C++ source code examples) which can be downloaded from http://www.st.com.

An executable with a simple user interface is also available for both Visual Basic and C/C++ projects.

Figure 1. DEMO-CR95HF-A evaluation board



For more information, refer to the following documents available on www.st.com:

- CR95HF: 13.56-MHz multi-protocol contactless transceiver IC with SPI and UART serial access datasheet
- DEMO-CR95HF-A: Demonstration kit for the CR95HF 13.56 MHz transceiver IC data brief
- UM1084: CR95HF development software user guide

Contents

1 Getting started 3

1.1 Connecting the board to your computer 3

1.2 Using the DLL system file 3

1.2.1 Overview of DLL functions 3

2 Function descriptions 6

2.1 Functions to check DLL installation and USB connection 6

2.1.1 CR95HFDll_GetDLLrev 6

2.1.2 CR95HFDLL_USBconnect 8

2.1.3 CR95HFDLL_USBhandlecheck 9

2.2 Functions to communicate with the STM32 MCU 11

2.2.1 CR95HFDll_Echo 11

2.2.2 CR95HFDLL_getMCUrev 13

2.2.3 CR95HFDLL_getInterfacePinState 14

2.3 Functions to communicate with the CR95HF IC 17

2.3.1 CR95HFDll_Idn 17

2.3.2 CR95HFDll_Select 19

2.3.3 CR95HFDll_STCmd 22

2.3.4 CR95HFDll_FieldOff 28

2.3.5 CR95HFDll_ResetSPI 29

2.3.6 CR95HFDll_SendIRQPulse 31

2.3.7 CR95HFDll_Polling_Reading 33

2.3.8 CR95HFDll_SendNSSPulse 34

2.4 Functions to communicate with a RF tag 36

2.4.1 CR95HFDll_SendReceive 36

Appendix A Error codes 40

Revision history 41

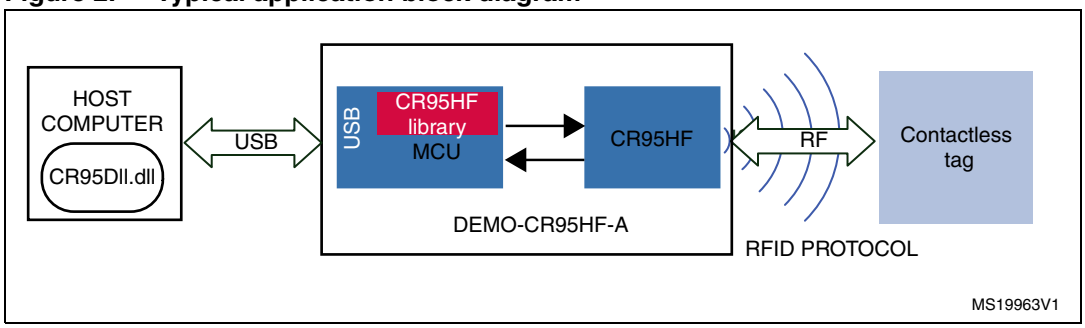
1 Getting started

1.1 Connecting the board to your computer

The DEMO-CR95HF-A is connected to the host computer through its USB port. No particular USB driver is needed. The HID USB driver (already included in all Windows operating systems) is used for communication.

Through the USB port of your computer, you can drive the board, communicate with the STM32 MCU, manage the CR95HF IC and send requests to a RF tag.

Figure 2. Typical application block diagram



1.2 Using the DLL system file

A DLL file (CR95HF.dll) is provided in order to drive the DEMO-CR95HF-A board from your computer.

Copy this file to the system folder of your computer (C:/Windows/system32/).

This DLL contains several functions to enable communication with the board.

Using this DLL, the host computer is able to manage several functions and communicate with the STM32 MCU and the CR95HF IC.

1.2.1 Overview of DLL functions

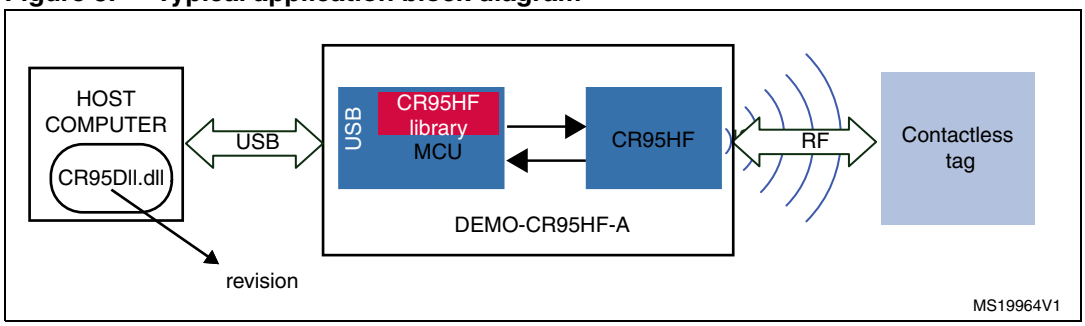
The *CR95HFDll_GetDLLrev* function reads the revision number of the DLL file. If the DLL file is not correctly installed in the system folder of the host computer, this function returns an error message (see *Figure 3*).

The *CR95HFDLL_USBconnect* function is required to connect the DEMO-CR95HF-A board to the host computer and communicate with it. It is available from DLL revision 0.5 and upper. The computer first checks if a DEMO-CR95HF-A is connected to the USB port. If so, it assigns a USB handle to the board (see *Figure 3*).

The *CR95HFDLL_USBhandlecheck* function can be used to verify if the DEMO-CR95HF-A is connected to the USB port of your computer. It is available from DLL revision 0.6 and upper. This function sends a request to the DEMO-CR95HF-A and waits for an acknowledge. An error code is sent if the communication with the board has been lost. In

this case, the CR95HFDLL_USBconnect can be used to reconnect the DEMO-CR95HF-A to the host computer.

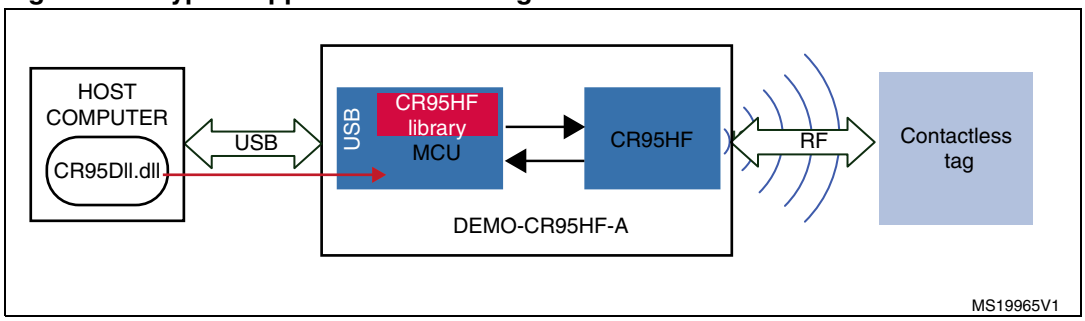
Figure 3. Typical application block diagram



A set of functions are available to drive the STM32 MCU (see *Figure 4*):

- *CR95HFDll_Echo*: verifies that the DEMO-CR95HF-A board is connected to the USB port of the computer.
- *CR95HFDLL_getMCUrev*: reads the firmware revision of the STM32 MCU.
- *CR95HFDLL_getInterfacePinState*: verifies the communication path between the STM32 MCU and the CR95HF IC (either SPI or UART).

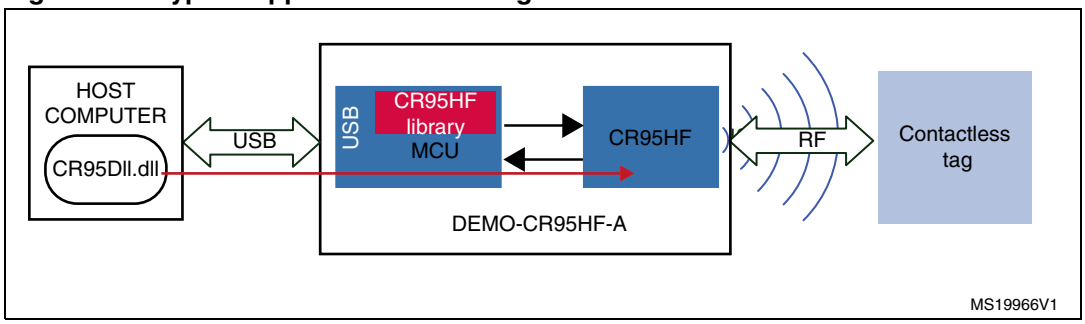
Figure 4. Typical application block diagram



Other functions can be used to send instructions to the CR95HF IC (see *Figure 5*):

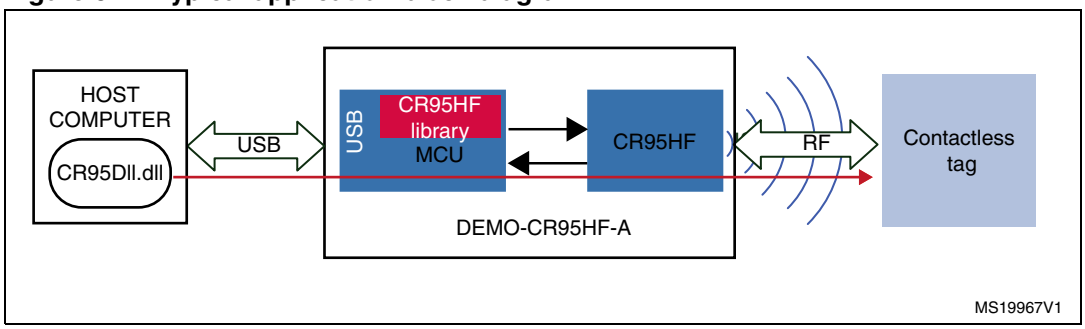
- *CR95HFDll_Idn*: gets the IDN of the CR95HF IC.
- *CR95HFDll_Select*: configures the required RF protocol for the CR95HF IC (IS0 15693, ISO 14443B …) and starts the RF field on the antenna.
- *CR95HFDll_STCmd*: sends commands to the CR95HF IC (refer to datasheet for a list of possible commands).
- *CR95HFDll_FieldOff*: stops the RF field of the CR95HF IC.
- *CR95HFDll_ResetSPI*: resets the SPI interface of the CR95HF IC.
- *CR95HFDll_SendIRQPulse*: sends an interruption pulse on the IRQ pin of the CR95HF IC to wake it up. This interrupt function can be used when the communication between the STM32 MCU and the DEMO-CR95HF-A is performed in SPI mode.
- *CR95HFDll_SendNSSPulse*: sends an interrupt pulse to the NSS pin of the CR95HF IC to wake it up. This interrupt function can be used when the communication between the STM32 MCU and the DEMO-CR95HF-A is performed in UART mode.
- *CR95HFDll_Polling_Reading*: launches a polling sequence and reads the CR95HF answer buffer to retrieve the answer when it is not available in the FIFO answer buffer (e.g. after an IDLE command)

Figure 5. Typical application block diagram



Finally, the *CR95HFDll_SendReceive* function sends an RF frame to any tag present in the RF field and receives its answer according to the RF protocol previously defined using the Select command (see *Figure 6*).

Figure 6. Typical application block diagram



2 Function descriptions

This section describes the various functions required to evaluate the DEMO-CR95HF-A demonstration board.

2.1 Functions to check DLL installation and USB connection

2.1.1 CR95HFDll_GetDLLrev

This function detects if the *CR95HF.dll* file has been installed in the system folder of the host PC system, reads its header and sends back its revision number.

This is used by software to ensure that the DLL is correctly installed and that it is up to date.

If no DLL file is found, it returns an error message.

Visual Basic language

Declaration:
```
Public Declare Function CR95HFDll_GetDLLrev Lib
"CR95HF.dll" (ByVal mystring As String) As Long
```

Prototype:
```
Dim lngStatus As Long
Dim strAnswer  As String * 256
lngStatus = CR95HFDll_GetDLLrev(strAnswer)
```

Input parameter: None

Output parameter: strAnswer: Revision of the DLL installed on your PC system.
Answer example: "0.4"

Returned value: lngStatus: 0

Source code example
```
Private Sub cmdGetDllRev_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long
lngStatus = CR95HFDll_GetDLLrev(strAnswer)
If (lngStatus = 0) Then
     txtDllRev.Text = strAnswer
Else
     txtDllRev.Text = "Error = no Dll found"
End If
End Sub
```

C/C++ language

Declaration: `__declspec(dllimport) int __stdcall CR95HFDll_GetDLLrev(char* stringreply);`

Prototype:
```
int iresult;
char strAnswer[50]="";
iresult = CR95HFDll_GetDLLrev(strAnswer);
```

Input parameter: None

Output parameter: strAnswer: Revision of the DLL installed on your PC system. Answer example: "0.4"

Returned value: Iresult: 0

Source code example

```
int Get_Dll_rev (void)
{
     char strAnswer[50]="";
     int iresult;
     char entry3;
     printf("\n\n\n\n");
     iresult = CR95HFDll_GetDLLrev(strAnswer);
     printf("\nGet Dll revision using CR95HFDll_GetDLLrev
     function:\n");
     printf("\n --> request : CR95HFDll_GetDLLrev(strAnswer)");
     printf("\n    <-- answer  : ");
     if (iresult == 0)
          printf("%s = DLL version\n\n\r", strAnswer);
     else
          printf("%s = Error : no communication with MCU\n\n\r",
          strAnswer);
     printf("\n\n\n\n");
     printf("\npress any key to continue");
     printf("\n");
     scanf("%s", &entry3);
     return iresult;
}
```

2.1.2 CR95HFDLL_USBconnect

This function can be used to connect the DEMO-CR95HF-A to the computer USB port. The computer then assigns an USB handle to the board.

Visual Basic language

Declaration:
```
Public Declare Function CR95HFDLL_USBconnect
Lib"CR95HF.dll" () As Long
```

Prototype:
```
Dim lngStatus As Long
lngStatus = CR95HFDLL_USBconnect()
```

Input parameter: None

Output parameter: None

Return value: lngStatus:

0: the DEMO-CR95HF is connected and ready to be used

1: DEMO-CR95HF-A board not connected

Source code example
```
Private Sub cmdUSBconnect_Click()
Dim lngStatus As Long
lngStatus = CR95HFDLL_USBconnect()
If (lngStatus = 0) Then
     txtUSBconnectAnswer.Text="DEMO-CR95HF-A connected and ready
     to be used"
Else
      txtUSBconnectAnswer.Text = "DEMO-CR95HF-A not connected"
End If
End Sub
```

C/C++ language

Declaration:
```
__declspec(dllexport) int  __stdcall
CR95HFDLL_USBconnect(void);
```

Prototype:
```
int iresult;
iresult= CR95HFDLL_USBconnect ();
```

Input parameter: None

Output parameter: None

Return value: iresult:

0: the DEMO-CR95HF is connected and ready to be used

1: DEMO-CR95HF-A board not connected

Source code example
```
int DEMO_CR95HF_A_USBconnect (void)
{
  int iresult;
  char entry3;
     printf("\n\n\n\n");
```

```
        iresult= CR95HFDLL_USBconnect ();
        printf("\nCR95HFDll_USBconnect function:\n");
        printf("\n    --> request : CR95HFDll_USBconnect()");
        printf("\n    <-- answer  : ");
        if (iresult == 0)
           printf("DEMO-CR95HF-A is connected and ready to be used\n\n\r");
        else
           printf("Error : DEMO-CR95HF-A not ready to be used\n\n\r");
        printf("\n\n\n\n");
        printf("\npress any key to continue");
        printf("\n");
        scanf("%s", &entry3);
        return iresult;
}
```

2.1.3 CR95HFDLL_USBhandlecheck

This function can be used to verify if the DEMO-CR95HF-A is correctly connected to the USB port of the computer and if the USB Handle attribué by the CR95HFDLL_USBconnect is always valid.

Visual Basic language

Declaration:
```
Public Declare Function CR95HFDLL_USBhandlecheck Lib
"CR95HF.dll" () As Long
```

Prototype:
```
Dim lngStatus as long
lngStatus = CR95HFDLL_USBhandlecheck()
```

Input parameter: None

Output parameter: None

Returned value: lngStatus:
0: the USB handle is valid
1: the USB handle is invalid and the DEMO-CR95HF-A board not connected

Source code example

```
Private Sub cmdUSBhandlecheck_Click()
Dim lngStatus As Long
txtUSBhandlecheckAnswer.Text = ""
lngStatus = CR95HFDLL_USBhandlecheck()
If (lngStatus = 0) Then
    txtUSBhandlecheckAnswer.Text = "DEMO-CR95HF-A connected"
ElseIf (lngStatus = 5) Then
    txtUSBhandlecheckAnswer.Text = "DEMO-CR95HF-A not connected (use
CR95HFDLL_USBconnect to connect it)"
End If

End Sub
```

C/C++ language

Declaration: `__declspec(dllexport) int __stdcall CR95HFDLL_USBhandcheck(void);`

Prototype: `int iresult;`
`iresult= CR95HFDLL_USBhandcheck();`

Input parameter: None

Output parameter: None

Return value: iresult:

0: the USB handle is valid
1: the USB handle is invalid and the DEMO-CR95HF-A board not connected

Source code example

```
int DEMO_CR95HF_A_USBhandlecheck (void)
{
  int iresult;
  char entry3;
  printf("\n\n\n\n");
  iresult= CR95HFDLL_USBhandlecheck ();
  printf("\nCR95HFDll_USBhandlecheck function:\n");
  printf("\n    --> request : CR95HFDll_USBhandlecheck()");
  printf("\n    <-- answer  : ");
  if (iresult == 0)
printf("DEMO-CR95HF-A is well connected\n\n\r");
  else
printf("Error : DEMO-CR95HF-A not connected (use
CR95HFDLL_USBConnect)\n\n\r");
  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);
  return iresult;
}
```

2.2 Functions to communicate with the STM32 MCU

2.2.1 CR95HFDll_Echo

This function sends a USB request to the STM32 MCU that executes an Echo request on the CR95HF. The STM32 MCU sends back the answer of the CR95HF, if possible, or returns an error code ‘1’ if there is no answer.

Note: The Echo request uses the EchoCode command to perform a serial interface echo. For more information, refer to the CR95HF transceiver datasheet.

Visual Basic language

Declaration:
```
Public Declare Function CR95HFDll_Echo Lib "CR95HF.dll"
(ByVal mystring As String) As Long
```

Prototype:
```
Dim lngStatus as long
Dim strAnswer  As String * 256
lngStatus = CR95HFDll_Echo(strAnswer)
```

Input parameter: None

Output parameter: strAnswer: The CR95HF IC answer to the Echo request is “5500” if there is no error.
Answer example: “5500”

Returned value: lngStatus:
0: No error
5: DEMO-CR95HF-A board not connected

Source code example
```
Private Sub cmdEcho_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long

lngStatus = CR95HFDll_Echo(strAnswer)

If (lngStatus = 0) Then
    txtEchoAnswer.Text = strAnswer
Else
    txtEchoAnswer.Text = "No communication with DEMO-CR95HF-A board"
End If

End Sub
```

C/C++ language

Declaration: `__declspec(dllimport) int __stdcall CR95HFDll_Echo(char *Stringreply);`

Prototype:
```
int iresult;
char strAnswer[50]="";
iresult= CR95HFDll_Echo (strAnswer);
```

Input parameter: None

Output parameter: strAnswer: The CR95HF IC answer to the Echo request is “5500” if there is no error.
Answer example: “5500”

Returned value: iresult:
- 0: No error
- 5: DEMO-CR95HF-A board not connected

Source code example

```
int Echo (void)
{
  char strAnswer[50]="";
  int iresult;
  char entry3;

  printf("\n\n\n\n");

  iresult= CR95HFDll_Echo (strAnswer);

  printf("\nEcho using CR95HFDll_Echo function:\n");
  printf("\n    --> request : CR95HFDll_Echo(strAnswer)");
  printf("\n    <-- answer  : ");

  if (iresult == 0)
          printf("%s = Echo command answer\n\n\r", strAnswer);
  else
          printf("%s = No answer from MCU\n\n\r", strAnswer);

  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

  return iresult;
}
```

2.2.2 CR95HFDLL_getMCUrev

This function sends a USB request to the STM32 MCU on the DEMO-CR95HF-A board that sends back the revision number of its firmware.

Visual Basic language

Declaration:
```
Public Declare Function CR95HFDLL_getMCUrev Lib
"CR95HF.dll" (ByVal mystring As String) As Long
```

Prototype:
```
status = CR95HFDLL_getMCUrev(strAnswer)
```

Input parameter: None

Output parameter: strAnswer: Firmware revision of the STM32 MCU on DEMO-CR95HF-A board.
Answer example: “0003010300”

Where:
- 00: Status byte (see *Appendix A* for error codes)
- 03: Size of answer (in bytes)
- 010300: Revision 1.3.0

Returned value: lngStatus:
- 0: No error
- 1: DEMO-CR95HF-A board not connected

Source code example
```
Private Sub cmdGetMCURev_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long

lngStatus = CR95HFDLL_getMCUrev(strAnswer)

If (lngStatus = 0) Then
    txtMCUrev.Text = strAnswer
Else
    txtMCUrev.Text = "Error = no communication with STM32 MCU"
End If

End Sub
```

C/C++ language

Declaration:
```
__declspec(dllexport) int  __stdcall
CR95HFDLL_getMCUrev(char* stringreply);
```

Prototype
```
int iresult;
char strAnswer[50]="";
iresult = CR95HFDLL_getMCUrev(strAnswer);
```

Input parameter: None

Output parameter: strAnswer: Firmware revision of the STM32 MCU on DEMO-CR95HF-A board.
Answer example: “0003010300”

Where:
- 00: Status byte (see *Appendix A* for error codes)

03: Size of answer (in bytes)
010300: Revision 1.3.0

Returned value: iresult:
0: No error
1: DEMO-CR95HF-A board not connected

Source code example

```
int Get_MCU_rev (void)
{
  char strAnswer[50]="";
  int iresult;
  char entry3;

  printf("\n\n\n\n");

  //iresult = CR95HFDLL_getMCUrev(strAnswer);

  printf("\nGet MCU revision using CR95HFDLL_getMCUrev function:\n");
  printf("\n    --> request : CR95HFDLL_getMCUrev(strAnswer)");
  printf("\n    <-- answer  : ");

  if (iresult == 0)
          printf("%s = MCU version\n\n\r", strAnswer);
  else
          printf("%s = Error : no communication with MCU\n\n\r",
strAnswer);

  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

  return iresult;
}
```

2.2.3 CR95HFDLL_getInterfacePinState

This function verifies the communication path between the STM32 MCU and the CR95HF IC (either SPI or UART).

The STM32 MCU checks which communication configuration is selected on the DEMO-CR95HF-A board. Two solder bridges are available to select the SPI or UART configuration. By defeat, the board is delivered in the SPI configuration.

Visual Basic language

Declaration: `Public Declare Function CR95HFDLL_getMCUrev Lib "CR95HF.dll" (ByVal mystring As String) As Long`

Prototype: `status = CR95HFDLL_getMCUrev(strAnswer)`

Input parameter: None

Output parameter: strAnswer: Interface Pin state.
Answer example: “00010X”

Where:
- 00: Status byte (see *Appendix A* for error codes)
- 01: Size of answer (in bytes)
- 0X: Communication mode

With X:
- 0: Communication in UART mode
- 1: Communication in SPI Mode

Returned value: lngStatus:
- 0: No error
- 5: DEMO-CR95HF-A board not connected

Source code example

```
Private Sub cmdGetInterfacePinState_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long

chkInterfacePinState(0).Value = vbUnchecked
chkInterfacePinState(1).Value = vbUnchecked

lngStatus = CR95HFDLL_getInterfacePinState(strAnswer)

If (lngStatus = 0) Then
    txtGetInterfacePinStateAnswer.Text = strAnswer
    If (Mid(strAnswer, 6, 1) = "0") Then
        chkInterfacePinState(0).Value = vbChecked
        chkInterfacePinState(1).Value = vbUnchecked
    ElseIf (Mid(strAnswer, 6, 1) = "1") Then
        chkInterfacePinState(0).Value = vbUnchecked
        chkInterfacePinState(1).Value = vbChecked
    Else
        DoEvents
        'error
    End If
Else
    txtGetInterfacePinStateAnswer.Text = "No answer from the DEMO-
CR95HF-A board"
End If

End Sub
```

C/C++ language

Declaration:

```
__declspec(dllexport) int  __stdcall
CR95HFDLL_getInterfacePinState(char* stringreply);
Prototypeint iresult;
char strAnswer[50]="";
iresult = CR95HFDLL_getInterfacePinState(strAnswer);
```

Input parameter: None

Output parameter: strAnswer: Interface Pin state.
Answer example: “00010X”

Where:
- 00: Status byte (see *Appendix A* for error codes)

01: Size of answer (in bytes)
0X: Communication mode

With X:
0: Communication in UART mode
1: Communication in SPI Mode

Returned value: iresult:
0: No error
5: DEMO-CR95HF-A board not connected

Source code example

```
int Get_Interface_Pin_State (void)
{
  int iresult;
  char strAnswer[50]="";
  char entry3;

  printf("\n\n\n\n");

  iresult = CR95HFDLL_getInterfacePinState(strAnswer);

  printf("\nGet Interface Pin State CR95HFDLL_getInterfacePinState
function:\n");
  printf("\n    --> request :
CR95HFDLL_getInterfacePinState(strAnswer)");
  printf("\n    <-- answer  : ");

  if (iresult == 0)
  {
       printf("%s = Get Interface Pin State successfully\n\n\r",
strAnswer);
       if (strAnswer[5] == '0')
          printf("\n        %c = communication in UART
mode",strAnswer[3]);
       else
          printf("\n        %c = communication in SPI
mode",strAnswer[3]);
  }
  else
       printf("\n        %s = No answer from DEMO-CR95HF-A\n\n\r",
strAnswer);

  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

  return iresult;
}
```

2.3 Functions to communicate with the CR95HF IC

2.3.1 CR95HFDll_Idn

This function sends a USB request to the STM32 MCU that requests the IDN of the CR95HF IC. The STM32 MCU send back the answer of the CR95HF containing the IDN value (ASCII codes), if possible, or returns an error code ‘1’ if there is no answer.

Note: The IDN command requests brief information about the CR95HF and its firmware version. For more information, refer to the CR95HF transceiver datasheet.

Visual Basic language

Declaration:
```
Public Declare Function CR95HFDll_Idn Lib "CR95HF.dll"
(ByVal mystring As String) As Long
```

Prototype:
```
Dim strAnswer  As String * 256
Dim lngStatus As Long
status = CR95HFDll_Idn(strAnswer)
```

Input parameter: None

Output parameter: strAnswer: IDN of the CR95HF IC (if no error)
Answer example: “000F4E4643204653324A4153543000A998”

Where:
- 00: Status byte (see annexe 2 for error codes)
- 0F: Size of answer (in bytes)
- 4E4643204653324A4153543000: ASCII transcription of the CR95HF IDN (in this example, IDN = “NFC FS2JAST0©”)
- A998: CRC value

Returned value: lngStatus:
- 0: No error
- 5: DEMO-CR95HF-A board not connected

Source code example

```
Private Sub cmdIDnRequest_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long

lngStatus = CR95HFDll_Idn(strAnswer)

If (lngStatus = 0) Then
    txtIdnAnswer.Text = strAnswer
Else
    txtIdnAnswer.Text = "Idn not sent back by the CR95HF IC"
End If

End Sub
```

C/C++ language

Declaration:
```
__declspec(dllimport) int  __stdcall CR95HFDll_Idn(char
*Stringreply);
Prototype int iresult;
```

```
char strAnswer[50]="";
iresult= CR95HFDll_Idn (strAnswer);
```

Input parameter: None

Output parameter: strAnswer: IDN of the CR95HF IC (if no error)
Answer example: “000F4E4643204653324A4153543000A998”

Where:
- 00: Status byte (see annexe 2 for error codes)
- 0F: Size of answer (in bytes)
- 4E4643204653324A4153543000: ASCII transcription of the CR95HF IDN (in this example, IDN = “NFC FS2JAST0©”)
- A998: CRC value

Returned value: iresult:
- 0: No error
- 5: DEMO-CR95HF-A board not connected

Source code example

```
int Idn (void)
{
  int iresult;
  char strAnswer[50]="";
  char entry3;

  printf("\n\n\n\n");

  iresult= CR95HFDll_Idn (strAnswer);

  printf("\nIdn using CR95HFDll_Idn function:\n");
  printf("\n    --> request : CR95HFDll_Idn(strAnswer)");
  printf("\n    <-- answer  : ");

  if (iresult == 0)
          printf("%s = Idn command response\n\n\r", strAnswer);
  else
          printf("%s = Error : no Idn returned\n\n\r", strAnswer);

  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

  return iresult;
}
```

2.3.2 CR95HFDll_Select

This function sends a USB request to the STM32 MCU that prepares the CR95HF for communication by executing a Select request containing the selected RF parameters to the the CR95HF IC. The STM32 MCU sends back the answer of the CR95HF, if possible, or returns an error code ‘1’ if there is no answer. In addition to selecting the correct RF communication parameters, this function activates the RF field. This is necessary at the start of communications if the RF field was previously switched off.

The parameters of this function indicate the selected RF communication protocol (ISO 15693, ISO 14443A, ISO 14443B, FELICA) as defined in the CR95HF datasheet.

Note: The Select request uses the Protocol Select command to select the RF communication protocol and specify certain protocol-related parameters. For more information, refer to the CR95HF transceiver datasheet.

Visual Basic language

Declaration:

```
Public Declare Function CR95HFDll_Select Lib
"CR95HF.dll" (ByVal mycmdstring As String, ByVal
mystring As String) As Long
```

Prototype:

```
Dim lngStatus As Long
Dim strConfiguration As String
Dim strSelectMsg  As String * 256
lngStatus = CR95HFDll_Select(strConfiguration,
strSelectMsg)
```

Input parameter: strConfiguration: Selected RF communication protocol and certain protocol-related parameters. (This configuration is used for SendReceive requests.)

ISO15693 HighDataRate 10% One subcarrier = “010D”:
Where:
“01” is the ISO 15693 configuration
“0D” are the parameters

ISO14443A = "0200":
Where:
“02” is the ISO 14443A configuration
“00” are the parameters

ISO14443B = "0301":
Where:
“03” is the ISO 14443B configuration
“01” are the parameters

Output parameter: strSelectMsg: The DEMO-CR95HF-A sends back an answer if the CR95HF is configured correctly and the RF field is on.
Answer example: “0000”

Where:
“00” is the status byte (see *Appendix A* for error codes)
“00” is the size of the answer

Returned value: lngStatus:
0: No error
5: DEMO-CR95HF-A board not connected
2: Empty argument error
3: Command parmeter error

Source code example

```
 'Private Sub cmdSelect_ISO15693_protocol_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long

lngStatus = CR95HFDll_Select("010D", strAnswer)

If (lngStatus = 0) Then
    txtISO15693_SelectAnswer.Text = strAnswer
Else
    txtISO15693_SelectAnswer.Text = " Select protocol ISO15693 not
executed"
End If

End Sub


Private Sub cmdSelect_ISO14443B_protocol_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long

lngStatus = CR95HFDll_Select("0301", strAnswer)

If (lngStatus = 0) Then
    txtISO14443B_SelectAnswer.Text = strAnswer
Else
    txtISO14443B_SelectAnswer.Text = "Select protocol ISO14443B not
executed"
End If

End Sub
```

C/C++ language

Declaration:

```
__declspec(dllimport) int  __stdcall
CR95HFDll_Select(char *StringCmd, char *Stringreply);
```

Prototype:

```
int iresult;
char strRequest[50]="";
char strAnswer[50]="";
iresult = CR95HFDll_Select(strRequest ,strAnswer);
```

Input parameter: strConfiguration: Selected RF communication protocol and certain protocol-related parameters. (This configuration is used for SendReceive requests.)

ISO15693 HighDataRate 10% One subcarrier = "010D":
Where:
“01” is the ISO 15693 configuration
“0D” are the parameters

ISO14443A = "0200":
Where:
“02” is the ISO 14443A configuration
“00” are the parameters

ISO14443B = "0301":
Where:
“03” is the ISO 14443B configuration
“01” are the parameters

Output parameter: strSelectMsg: The DEMO-CR95HF-A sends back an answer if the CR95HF is configured correctly and the RF field is on.
Answer example: “0000”

Where:
“00” is the status byte (see *Appendix A* for error codes)
“00” is the size of the answer

Returned value: iresult:
0: No error
5: DEMO-CR95HF-A board not connected
2: Empty argument error
3: Command parmeter error

Source code example

```
int Select_ISO15693 (void)
{
  int iresult;
  char strRequest[50]="";
  char strAnswer[50]="";

  char entry3;

  printf("\n\n\n\n");

  strcpy(strRequest,"010D");

  iresult = CR95HFDll_Select(strRequest ,strAnswer);

  printf("\nProtocol select using CR95HFDll_Select function:\n");
  printf("\n    --> request : CR95HFDll_Select(%s, strAnswer)",
strRequest);
  printf("\n    <-- answer  : ");

  if (iresult == 0)
          printf("%s = ISO15693 protocol selected\n\n\r", strAnswer);
  else
          printf("%s = Select command error\n\n\r", strAnswer);

  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

  return iresult;
}
```

```
int Select_ISO14443B (void)
{
  char strRequest[50]="";
  char strAnswer[50]="";
  int iresult;
  char entry3;

  printf("\n\n\n\n");

  strcpy(strRequest,"0301");

  iresult = CR95HFDll_Select(strRequest ,strAnswer);

  printf("\nISO14443B protocol select using CR95HFDll_Select
function:\n");
  printf("\n    --> request : CR95HFDll_Select(%s,
strAnswer)",strRequest);
  printf("\n    <-- answer  : ");

  if (iresult == 0)
        printf("%s = ISO14443B protocol selected\n\n\r", strAnswer);
  else
        printf("%s = Select command error\n\n\r", strAnswer);

  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

  return iresult;
}
```

2.3.3 CR95HFDll_STCmd

This function is used to send any request to the CR95HF IC. The STM32 MCU receives the frame contained in the request and sends it directly to the CR95HF IC. The STM32 MCU sends back the answer from the CR95HF IC to the PC through the USB port. The frame has to be formatted according to the CR95HF datasheet.

This function can be defined as a “Transparent Mode” command.

The CR95HF formatted frame has to be send through this CR95HFDll_STCmd function preceded by “01” data as the header byte.

Visual Basic language

Declaration:

```
Public Declare Function CR95HFDll_STCmd Lib "CR95HF.dll"
(ByVal mycmdstring As String, ByVal mystring As String)
As Long
```

Prototype:

```
Dim lngStatus As Long
Dim strMCUrequest As String
Dim strCR95HFanswer As String * 256
```

```
lngStatus = CR95HFDll_STCmd(strMCUrequest,
strCR95HFanswer)
```

Input parameter: strMCUrequest: request to be sent by the STM32 MCU to the CR95HF IC, preceded by “01”.

Example 1: Send RdReg (Read Wakeup Register) command to read CR95HF wakeup flags: “010803620100”

Where:

“01” is the transparent command header byte
“0803620100” is the request to be send to the CR95HF IC
“08” is the RdReg command
“03” is the request length
“62” is the wakeup register value
“01” is the register size
“00” ST reserved

Example 2: Select ISO 15693 protocol: “010202010D”

Where:

“01” is the transparent command header byte
“0202010D” is the Protocol Select ISO 15693 frame
“02” is the Protocol Select command
“02” is the request length
“010D” are the Protocol Select command parameters

Example 3: IDLE mode:
“01070E03A200F80198000160600000 3F28”

Where:

“01” is the transparent command header byte
“070E03A200F80198000160600000 3F28” is the Idle frame for Tag Calibration

Output parameter: strCR95HFanswer: Answer of the CR95HF IC

Example 1: Answer to RdReg command = “000102”

Where:

“00” is the status byte (see *Appendix A* for error codes)
“01” is the size of the answer
“02” is the register value

Example 2: Answer to Select ISO 15693 protocol = “0000”

Where:

“00” is the status byte (see *Appendix A* for error codes)
“00” is the size of the answer

Example 3: Answer to Idle command = “0000”

Where:
"00" is the status byte (see *Appendix A* for error codes)
"00" is the size of the answer

Returned value: lngStatus:
0: No error
5: DEMO-CR95HF-A board not connected
2: Empty argument error

Source code example

```
Private Sub cmdSTCmd_RdReg_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long

lngStatus = CR95HFDll_STCmd("010803620100", strAnswer)

If (lngStatus = 0) Then
    txtSTCmd_RdReg.Text = strAnswer
Else
    txtSTCmd_RdReg.Text = "Read register not executed properly"
End If

End Sub


Private Sub cmdSTCmd_ISO15693_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long

lngStatus = CR95HFDll_STCmd("010202010D", strAnswer)

If (lngStatus = 0) Then
    txtSTCmd_ISO15693.Text = strAnswer
Else
    txtSTCmd_ISO15693.Text = " ISO15693 protocol not selected"
End If

End Sub


Private Sub cmdSTCmd_Idle_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long

lngStatus = CR95HFDll_STCmd("01070E03A200F80198000160600000 3F28",
strAnswer)

If (lngStatus = 0) Then
    txtSTCmd_Idle.Text = strAnswer
Else
    txtSTCmd_Idle.Text = " No Idn returned from CR95HF IC "
End If

End Sub
```

C/C++ language

Declaration:

```
__declspec(dllimport) int  __stdcall
CR95HFDll_STCmd(char *StringCmd, char *StringReply);
```

Prototype:

```
int iresult;
char strRequest[50]="";
char strAnswer[50]="";
iresult = CR95HFDll_STCmd(strRequest ,strAnswer);
```

Input parameter: strMCUrequest: Request to be sent by the STM32 MCU to the CR95HF IC following the Datasheet.

Example 1: Send RdReg (Read Wakeup Register) command to read CR95HF wakeup flags: “010803620100”

Where:

“01” is the transparent command header byte
“0803620100” is the request to be send to the CR95HF IC
“08” is the RdReg command
“03” is the request length
“62” is the wakeup register value
“01” is the register size
“00” ST reserved

Example 2: Select ISO 15693 protocol: “010202010D”

Where:

“01” is the transparent command header byte
“0202010D” is the Protocol Select ISO 15693 frame
“02” is the Protocol Select command
“02” is the request length
“010D” are the Protocol Select command parameters

Example 3: IDLE mode:
“01070E03A200F80198000160600000 3F28”

Where:

“01” is the transparent command header byte
“070E03A200F80198000160600000 3F28” is the Idle frame for Tag Calibration

Output parameter: strCR95HFanswer: Answer of the CR95HF IC

Example 1: Answer to RdReg command = “000102”

Where:

“00” is the status byte (see *Appendix A* for error codes)
“01” is the size of the answer
“02” is the register value

Example 2: Answer to Select ISO 15693 protocol = “0000”

Where:

“00” is the status byte (see *Appendix A* for error codes)
“00” is the size of the answer

Example 3: Answer to Idle command = “0000”

Where:

“00” is the status byte (see *Appendix A* for error codes)
“00” is the size of the answer

Returned value: iresult:

0: No error
5: DEMO-CR95HF-A board not connected
2: Empty argument error

Source code example

```
int STCmd_RdReg (void)
{
  int iresult;
  char strRequest[50]="";
  char strAnswer[50]="";
  char entry3;

  printf("\n\n\n\n");

  strcpy(strRequest,"010803620100");

  iresult = CR95HFDll_STCmd(strRequest ,strAnswer);

  printf("\nRead register using CR95HFDll_STCmd function:\n");
  printf("\n    --> request : CR95HFDll_STCmd(%s,
strAnswer)",strRequest);
  printf("\n    <-- answer  : ");

  if (iresult == 0)
          printf("%s = Read Register answer\n\n\r", strAnswer);
  else
          printf("%s = No answer from CR95HF IC\n\n\r", strAnswer);

  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

 return iresult;
}



int STCmd_ISO15693 (void)
{
  char strRequest[50]="";
  char strAnswer[50]="";
  int iresult;
```

```
  char entry3;

  printf("\n\n\n\n");

  strcpy(strRequest,"010202010D");

  iresult = CR95HFDll_STCmd(strRequest ,strAnswer);

  printf("\nISO15693 protocol select using CR95HFDll_STCmd
function:\n");
  printf("\n    --> request : CR95HFDll_STCmd(%s,
strAnswer)",strRequest);
  printf("\n    <-- answer  : ");

  if (iresult == 0)
          printf("%s = ISO15693 protocol selected successfully\n\n\r",
strAnswer);
  else
          printf("%s = Protocol not selected\n\n\r", strAnswer);

  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

  return iresult;
}



int STCmd_Idle (void)
{
  char strRequest[50]="";
  char strAnswer[50]="";
  int iresult;
  char entry3;

  printf("\n\n\n\n");

  strcpy(strRequest,"01070E03A200F801980001606000003F28");

  iresult = CR95HFDll_STCmd(strRequest ,strAnswer);

  printf("\nIdle using CR95HFDll_STCmd function:\n");
  printf("\n    --> request : CR95HFDll_STCmd(%s,
strAnswer)",strRequest);
  printf("\n    <-- answer  : ");

  if (iresult == 0)
          printf("%s = Idle answer\n\n\r", strAnswer);
  else
          printf("%s = No answer from CR95HF IC\n\n\r", strAnswer);

  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

  return iresult;
}
```

2.3.4 CR95HFDll_FieldOff

This function sends a USB request to the STM32 MCU to switch off the CR95HF RF Field. The STM32 MCU sends back the answer of the CR95HF, if possible, or returns an error code ‘1’ if there is no answer.

Note: To perform this RF Field Off command, the STM32 MCU sends a Protocol Select command with parameters ‘0000’. For more information, refer to the CR95HF transceiver datasheet.

Visual Basic language

Declaration:
```
Public Declare Function CR95HFDll_FieldOff Lib
"CR95HF.dll" (ByVal mystring As String) As Long
```

Prototype:
```
Dim lngStatus As Long
Dim strAnswer  As String * 256
status = CR95HFDll_FieldOff(strAnswer)
```

Input parameter: None

Output parameter: strAnswer: The DEMO-CR95HF-A sends back an answer and the RF Field is switched off.

Answer example: “0000”
Where:
“00” is the status byte (see *Appendix A* for error codes)
“00” is the size of the answer

Returned value: lngStatus:
0: No error
5: DEMO-CR95HF-A board not connected

Source code example
```
Private Sub cmdFieldOff_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long

lngStatus = CR95HFDll_FieldOff(strAnswer)

If (lngStatus = 0) Then
    txtFieldOffAnswer.Text = strAnswer
Else
    txtFieldOffAnswer.Text = "Field Off request not executed"
End If

End Sub
```

C/C++ language

Declaration:
```
__declspec(dllimport) int  __stdcall
CR95HFDll_FieldOff(char *StringReply);
```

Prototype:
```
int iresult;
char strAnswer[50]="";
iresult= CR95HFDll_FieldOff (strAnswer);
```

Input parameter: None

Output parameter: strAnswer: The DEMO-CR95HF-A sends back an answer and the RF Field is switched off.

Answer example: “0000”
Where:
“00” is the status byte (see *Appendix A* for error codes)
“00” is the size of the answer

Returned value: iresult:
0: No error
5: DEMO-CR95HF-A board not connected

Source code example

```
int FieldOff (void)
{
  int iresult;
  char strAnswer[50]="";
  char entry3;

  printf("\n\n\n\n");

  iresult= CR95HFDll_FieldOff (strAnswer);

  printf("\nField Off using CR95HFDll_FieldOff function:\n");
  printf("\n    --> request : CR95HFDll_FieldOff(strAnswer)");
  printf("\n    <-- answer  : ");

  if (iresult == 0)
          printf("%s = RF Field Off ok\n\n\r", strAnswer);
  else
          printf("%s = Error : RF Field Off command error\n\n\r",
strAnswer);

  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

  return iresult;
}
```

2.3.5 CR95HFDll_ResetSPI

This function resets the CR95HF IC in case of a problem. This function only resets the CR95HF IC and not the STM32 MCU.

Visual Basic language

Declaration:

```
Public Declare Function CR95HFDll_ResetSPI Lib
"CR95HF.dll" (ByVal mystring As String) As Long
```

Prototype:

```
Dim lngStatus As Long
Dim strAnswer  As String * 256
lngStatus = CR95HFDll_ResetSPI(strAnswer)
```

Input parameter: None

Output parameter: strAnswer: The DEMO-CR95HF-A sends back an answer if the SPI has been correctly reset.

Answer example: “0000”

Where:
“00” is the status byte (see *Appendix A* for error codes)
“00” is the size of the answer

Returned value: lngStatus:
0: No error
5: DEMO-CR95HF-A board not connected

Source code example

```
Private Sub cmdResetSPI_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long

lngStatus = CR95HFDll_ResetSPI(strAnswer)

If (lngStatus = 0) Then
    txtResetSPIAnswer.Text = strAnswer
Else
    txtResetSPIAnswer.Text = "No answer from the DEMO-CR95HF-A board"
End If

End Sub
```

C/C++ language

Declaration:

```
__declspec(dllexport) int  __stdcall
CR95HFDll_ResetSPI(char* stringreply);
```

Prototype:

```
int iresult;
char strAnswer[50]="";
iresult = CR95HFDll_ResetSPI(strAnswer);
```

Input parameter: None

Output parameter: strAnswer: The DEMO-CR95HF-A sends back an answer if the SPI has been correctly reset.

Answer example: “0000”

Where:
“00” is the status byte (see *Appendix A* for error codes)
“00” is the size of the answer

Returned value: iresult:
0: No error
5: DEMO-CR95HF-A board not connected

Source code example

```
int ResetSPI (void)
{
  int iresult;
  char strAnswer[50]="";
  char entry3;

  printf("\n\n\n\n");
```

```
  iresult = CR95HFDll_ResetSPI(strAnswer);

  printf("\nReset SPI using CR95HFDll_ResetSPI function:\n");
  printf("\n    --> request : CR95HFDll_ResetSPI(strAnswer)");
  printf("\n    <-- answer  : ");

  if (iresult == 0)
         printf("%s = Reset SPI successfully\n\n\r", strAnswer);
  else
         printf("%s = No answer from DEMO-CR95HF-A\n\n\r", strAnswer);

  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

  return iresult;
}
```

2.3.6 CR95HFDll_SendIRQPulse

This function must be used when the DEMO-CR95HF-A is configured in SPI mode (communication between the STM32 MCU and the CR95HF IC). The interrupt pulse is sent to the CR95HF IRQ pin.

Visual Basic language

Declaration:
```
Public Declare Function CR95HFDll_SendIRQPulse Lib
"CR95HF.dll" (ByVal mystring As String) As Long
```

Prototype:
```
Dim lngStatus As Long
Dim strAnswer  As String * 256
lngStatus = CR95HFDll_SendIRQPulse(strAnswer)
```

Input parameter: None

Output parameter: strAnswer: The DEMO-CR95HF-A sends back an answer if the IRQ pulse was correctly sent.

Answer example: “0000”

Where:
- “00” is the status byte (see *Appendix A* for error codes)
- “00” is the size of the answer

Returned value: lngStatus:
- 0: No error
- 5: DEMO-CR95HF-A board not connected

Source code example

```
Private Sub cmdSendIRQPulse_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long

lngStatus = CR95HFDll_SendIRQPulse(strAnswer)

If (lngStatus = 0) Then
```

```
    txtSendIRQPulseAnswer.Text = strAnswer
Else
    txtSendIRQPulseAnswer.Text = "No answer from the DEMO-CR95HF-A
board"
End If

End Sub
```

C/C++ language

Declaration:

```
__declspec(dllexport) int  __stdcall
CR95HFDll_SendIRQPulse(char* stringreply);
```

Prototype:

```
int iresult;
char strAnswer[50]="";
iresult = CR95HFDll_SendIRQPulse(strAnswer);
```

Input parameter: None

Output parameter: strAnswer: The DEMO-CR95HF-A sends back an answer if the IRQ Pulse was correctly sent.

Answer example: “0000”

Where:

“00” is the status byte (see *Appendix A* for error codes)
“00” is the size of the answer

Returned value: iresult:

0: No error
5: DEMO-CR95HF-A board not connected

Source code example

```
int Send_IRQ_Pulse (void)
{
  int iresult;
  char strAnswer[50]="";
  char entry3;

  printf("\n\n\n\n");

  //iresult = CR95HFDll_SendIRQPulse(strAnswer);

  printf("\nSend IRQ Pulse using CR95HFDll_SendIRQPulse function:\n");
  printf("\n    --> request : CR95HFDll_SendIRQPulse(strAnswer)");
  printf("\n    <-- answer  : ");

  if (iresult == 0)
          printf("%s = Send IRQ pulse successfully\n\n\r", strAnswer);
  else
          printf("%s = No answer from DEMO-CR95HF-A\n\n\r", strAnswer);

  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

  return iresult;
}
```

2.3.7 CR95HFDll_Polling_Reading

This function polls the CR95HF SPI communication wires. When the CR95HF is ready, it sends back the data stored in its FIFO answer buffer.

This function can be used after an IDLE command to make sure that the answer buffer has been emptied and avoid receiving an erroneous answer on the next request.

Visual Basic language

Declaration:
```
Public Declare Function CR95HFDll_Polling_Reading Lib
"CR95HF.dll" (ByVal mystring As String) As Long
```

Prototype:
```
Dim strAnswer  As String * 256
Dim lngStatus As Long
lngStatus = CR95HFDll_Polling_Reading(strAnswer)
```

Input parameter: None

Output parameter: CR95HF answer stored in FIFO answer buffer

Return value: lngStatus:

0: No error

5: DEMO-CR95HF-A board not connected

Source code example
```
Private Sub cmdPollingReading_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long
lngStatus = CR95HFDll_Polling_Reading(strAnswer)
If (lngStatus = 0) Then
    txtPollingReadingAnswer.Text = strAnswer
Else
    txtPollingReadingAnswer.Text = "No answer from the
   DEMO-CR95HF-A board"
End If
End Sub
```

C/C++ language

Declaration:
```
__declspec(dllexport) int  __stdcall
CR95HFDll_Polling_Reading(char* StringReply);
```

Prototype:
```
int iresult;
char strAnswer[50]="";
iresult = CR95HFDll_Polling_Reading(strAnswer);
```

Input parameter: None

Output parameter: CR95HF answer stored in FIFO answer buffer

Return value: iresult:

0: No error
5: DEMO-CR95HF-A board not connected

Source code example
```
int Set_Polling_Reading_config (void)
{
```

```
int iresult;
char strAnswer[50]="";
char entry3;
printf("\n\n\n\n");
iresult = CR95HFDll_Polling_Reading(strAnswer);
printf("\nSend IRQ Pulse using CR95HFDll_Polling_Readin
function:\n");
printf("\n    --> request : CR95HFDll_Polling_Reading(strAnswer)");
printf("\n    <-- answer  : ");
if (iresult == 0)
   printf("%s = Set Polling Reading successfully\n\n\r", strAnswer);
else
   printf("%s = No answer from DEMO-CR95HF-A\n\n\r", strAnswer);
printf("\n\n\n\n");
printf("\npress any key to continue");
printf("\n");
scanf("%s", &entry3);
return iresult;
}
```

2.3.8 CR95HFDll_SendNSSPulse

This function sends an interrupt to wake up the CR95HF IC. It can be used when the DEMO-CR95HF-A is configured in UART mode (communication between the STM32 MCU and the CR95HF IC). The interrupt pulse is sent to the CR95HF NSS pin.

Visual Basic language

Declaration:

```
Public Declare Function CR95HFDll_SendNSSPulse Lib
"CR95HF.dll" (ByVal mystring As String) As Long
```

Prototype:

```
Dim strAnswer  As String * 256
Dim lngStatus As Long
lngStatus = CR95HFDll_SendNSSPulse(strAnswer)
```

Input parameter: None

Output parameter: strAnswer. The DEMO-CR95HF-A sends back an answer if the NSS pulse was correctly sent.

Example of answer: “0000”
where
“00” is the status byte (see Appendix A for error codes)
“00” is the size of the answer

Return value: lngStatus:

0: No error
5: DEMO-CR95HF-A board not connected

Source code example

```
Private Sub cmdSendNSSPulse_Click()
Dim strAnswer  As String * 256
Dim lngStatus As Long
lngStatus = CR95HFDll_SendNSSPulse(strAnswer)
If (lngStatus = 0) Then
    txtNSSpulseAnswer.Text = strAnswer
Else
    txtNSSpulseAnswer.Text = "No answer from the DEMO-CR95HF-A board"
End If
End Sub
```

C/C++ language

Declaration:

```
__declspec(dllexport) int  __stdcall
CR95HFDll_SendNSSPulse(char* StringReply);
```

Prototype:

```
int iresult;
char strAnswer[50]="";
iresult = CR95HFDll_SendNSSPulse(strAnswer);
```

Input parameter: None

Output parameter: strAnswer: The DEMO-CR95HF-Asends back an answer if the NSS pulse was correctly sent.
Example of answer: “0000”
where
“00” is the status byte (see Appendix A for error codes)
“00” is the size of the answer

Return value: iresult:

0: No error
5: DEMO-CR95HF-A board not connected

Source code example

```
int Send_NSS_Pulse (void)
{
  int iresult;
  char strAnswer[50]="";
  char entry3;
  printf("\n\n\n\n");
  iresult = CR95HFDll_SendNSSPulse(strAnswer);
  printf("\nSend IRQ Pulse using CR95HFDll_SendNSSPulse function:\n");
  printf("\n    --> request : CR95HFDll_SendNSSPulse(strAnswer)");
  printf("\n    <-- answer  : ");
  if (iresult == 0)
         printf("%s = Send NSS pulse successfully\n\n\r", strAnswer);
  else
          printf("%s = No answer from DEMO-CR95HF-A\n\n\r", strAnswer);
  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);
  return iresult;
}
```

2.4 Functions to communicate with a RF tag

2.4.1 CR95HFDll_SendReceive

This function sends a USB request to the STM32 MCU that executes a SendRecv command with data to the CR95HF IC. The STM32 MCU sends back the answer of the CR95HF, if possible, or returns an error code ‘1’ if there is no answer.

Note: The request uses the SendRecv command to send data using previously selected protocol and to receive the tag response. For more information, refer to the CR95HF transceiver datasheet.

Visual Basic language

Declaration:

```
Public Declare Function CR95HFDll_SendReceive Lib
"CR95HF.dll" (ByVal mycmdstring As String, ByVal
mystring As String) As Long
```

Prototype:

```
Dim lngStatus As Long
Dim strCR95HFrequest As String
Dim strTagResponse  As String * 256
lngStatus = CR95HFDll_SendReceive(strCR95HFrequest,
strTagResponse  )
```

Input parameter: strCR95HFrequest: The RF Request to be sent by the CR95HF IC to the Tag (with previously selected ISO format).

ISO 15693 Inventory example: “260100”

Where:
“260100” is the ISO 15693 Inventory command.

ISO 14443B Initiate example: "0600"

Where:
“0600” is the ISO 14443B Initiate command.

Output parameter: strTagResponse: The Tag answer if the CR95HF has received an answer from the Tag in the field; otherwise, an error code (see *Appendix A*).

ISO 15693 Inventory example:
“800D00FF5AB914F6414C02E0385D00”

Where:
“80” is the status byte (see *Appendix A* for error codes)
“0D” is the length of entire data field
“00FF5AB914F6414C02E0” data received from the tag
“385D” is the original received CRC value
“00” is the protocol error status

ISO 14443B Initiate example: “80046C125900”

Where:
“80” is the status byte (see annexe 2 for error codes)
“04” is the length of entire data field
“6C” is the data received from the tag

"1259" is the original received CRC value
"00" is the protocol error status

Returned value: lngStatus:
0: No error
5: DEMO-CR95HF-A board not connected
4: Communication error

Source code example

```
Private Sub cmdSendReceive_ISO15693_Inventory_Click()
Dim strTagResponse  As String * 256
Dim lngStatus As Long

lngStatus = CR95HFDll_SendReceive("260100", strTagResponse)

If (lngStatus = 0) Then
    txtISO15693_Inventory_Answer.Text = strTagResponse
Else
    txtISO15693_Inventory_Answer.Text = "No answer from the Tag"
End If

End Sub


Private Sub cmdSendReceive_ISO14443B_Initiate_Click()
Dim strTagResponse  As String * 256
Dim lngStatus As Long

lngStatus = CR95HFDll_SendReceive("0600", strTagResponse)

If (lngStatus = 0) Then
    txtISO14443B_Initiate_Answer.Text = strTagResponse
Else
    txtISO14443B_Initiate_Answer.Text = "No answer from the Tag"
End If

End Sub
```

C/C++ language

Declaration:

```
__declspec(dllimport) int  __stdcall
CR95HFDll_SendReceive(char *StringCmd, char
*StringReply);
```

Prototype:

```
int iresult;
char strRequest[50]="";
char strTagAnswer[50]="";
iresult=CR95HFDll_SendReceive(strRequest,strTagAnswer);
```

Input parameter: strCR95HFrequest: The RF Request to be sent by the CR95HF IC to the Tag (with previously selected ISO format).

ISO 15693 Inventory example: "260100"

Where:
"260100" is the ISO 15693 Inventory command.

ISO 14443B Initiate example: "0600"

Where:
“0600” is the ISO 14443B Initiate command.

Output parameter: strTagResponse: The Tag answer if the CR95HF has received an answer from the Tag in the field; otherwise, an error code (see *Appendix A*).

ISO 15693 Inventory example:
“800D00FF5AB914F6414C02E0385D00”

Where:
“80” is the status byte (see *Appendix A* for error codes)
“0D” is the length of entire data field
“00FF5AB914F6414C02E0” data received from the tag
“385D” is the original received CRC value
“00” is the protocol error status

ISO 14443B Initiate example: “80046C125900”

Where:
“80” is the status byte (see annexe 2 for error codes)
“04” is the length of entire data field
“6C” is the data received from the tag
“1259” is the original received CRC value
“00” is the protocol error status

Returned value: iresult:
0: No error
5: DEMO-CR95HF-A board not connected
4: Communication error

Source code example

```
int Send_ISO15693_Inventory (void)
{
  int iresult;
  char strRequest[50]="";
  char strTagAnswer[50]="";
  char entry3;

  printf("\n\n\n\n");

  strcpy(strRequest,"260100");

  iresult=CR95HFDll_SendReceive(strRequest,strTagAnswer);

  printf("\nISO15695 inventory using CR95HFDll_SendReceive
function:\n");
  printf("\n    --> request : CR95HFDll_SendReceive(%s,
strTagAnswer)",strRequest);
  printf("\n    <-- answer  : ");

  if ((strTagAnswer[0] == '8') & (strTagAnswer[1] == '0')) //CR95HF Tag
answer OK
  {
     printf("%s = Tag answer",strTagAnswer);
  }
  else
  {
     printf("%s = No tag answer received\n");
```

```
  }

  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

  return iresult;
}


int Send_ISO14443B_Initiate (void)
{
  char strRequest[50]="";
  char strTagAnswer[50]="";
  int iresult;
  char entry3;

  printf("\n\n\n\n");

  strcpy(strRequest,"0600");

  iresult=CR95HFDll_SendReceive(strRequest,strTagAnswer);

  printf("\nISO14443B Initiate using CR95HFDll_SendReceive
function:\n");
  printf("\n    --> request : CR95HFDll_SendReceive(%s,
strTagAnswer)",strRequest);
  printf("\n    <-- answer  : ");

  if ((strTagAnswer[0] == '8') & (strTagAnswer[1] == '0')) //CR95HF Tag
answer OK
  {
     printf("%s = Tag answer",strTagAnswer);
  }
  else
  {
     printf("No tag answer received\n");
     printf("CR95HF answer : %s",strTagAnswer);
  }
  printf("\n\n\n\n");
  printf("\npress any key to continue");
  printf("\n");
  scanf("%s", &entry3);

  return iresult;
}
```

Appendix A Error codes

Table 1. List of error codes

Error code	Description
0000	Answer OK
8000	Answer OK
8200	Invalid command length
8300	Invalid protocol
8600	Communication error
8700	Frame wait time out OR no tag
8800	Invalid Start Of Frame
8900	Receive buffer overflow (too many bytes received)
8A00	Framing error (start bit = 0, stop bit = 1)
8B00	EGT time out (for ISOIEC 14443-B)
8C00	Invalid length. Used in Felica, when field length < 3
8D00	CRC error (Used in Felica protocol)
8E00	Reception lost without EOF received
FD00	Time out - no answer from Tag detected by the CR95HF IC
FE00	Unknown error

Revision history

Table 2. Document revision history

Date	Revision	Changes
25-Jul-2011	1	Initial release.
16-Dec-2011	2	Added functions available in new DLL release revision 0.5: *Section 2.1.2: CR95HFDLL_USBconnect*, *Section 2.1.3: CR95HFDLL_USBhandlecheck*, *Section 2.3.7: CR95HFDll_Polling_Reading*, and *Section 2.3.8: CR95HFDll_SendNSSPulse*. Updated returned value in C/C++ for *Section 2.1.1: CR95HFDll_GetDLLrev*, *Section 2.2.2: CR95HFDLL_getMCUrev*, *Section 2.2.3: CR95HFDLL_getInterfacePinState*, *Section 2.3.1: CR95HFDll_Idn*, *Section 2.3.2: CR95HFDll_Select*, *Section 2.3.3: CR95HFDll_STCmd*, *Section 2.3.4: CR95HFDll_FieldOff*, *Section 2.3.5: CR95HFDll_ResetSPI*, *Section 2.3.6: CR95HFDll_SendIRQPulse*, *Section 2.4.1: CR95HFDll_SendReceive*. Updated returned value in Visual basic and C/C++ for *Section 2.2.1: CR95HFDll_Echo*.

Please Read Carefully:

Information in this document is provided solely in connection with ST products. STMicroelectronics NV and its subsidiaries (“ST”) reserve the right to make changes, corrections, modifications or improvements, to this document, and the products and services described herein at any time, without notice.

All ST products are sold pursuant to ST’s terms and conditions of sale.

Purchasers are solely responsible for the choice, selection and use of the ST products and services described herein, and ST assumes no liability whatsoever relating to the choice, selection or use of the ST products and services described herein.

No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services it shall not be deemed a license grant by ST for the use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering the use in any manner whatsoever of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN ST’S TERMS AND CONDITIONS OF SALE ST DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO THE USE AND/OR SALE OF ST PRODUCTS INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

UNLESS EXPRESSLY APPROVED IN WRITING BY TWO AUTHORIZED ST REPRESENTATIVES, ST PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN MILITARY, AIR CRAFT, SPACE, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE. ST PRODUCTS WHICH ARE NOT SPECIFIED AS "AUTOMOTIVE GRADE" MAY ONLY BE USED IN AUTOMOTIVE APPLICATIONS AT USER’S OWN RISK.

Resale of ST products with provisions different from the statements and/or technical features set forth in this document shall immediately void any warranty granted by ST for the ST product or service described herein and shall not create or extend in any manner whatsoever, any liability of ST.

ST and the ST logo are trademarks or registered trademarks of ST in various countries.

Information in this document supersedes and replaces all information previously supplied.

The ST logo is a registered trademark of STMicroelectronics. All other names are the property of their respective owners.

© 2011 STMicroelectronics - All rights reserved

STMicroelectronics group of companies

Australia - Belgium - Brazil - Canada - China - Czech Republic - Finland - France - Germany - Hong Kong - India - Israel - Italy - Japan - Malaysia - Malta - Morocco - Philippines - Singapore - Spain - Sweden - Switzerland - United Kingdom - United States of America

www.st.com